Amendments to the Drawings:

The attached replacement sheets of drawings includes changes to Figs. 1, 2 and 3 and replaces the original sheets including Figs.1, 2 and 3.

In Figure 1, the applicant changed the indicated name of the component marked 26a to "SDRAM Controller", and changed the indicated name of the component marked 26b to "SRAM Controller." Additionally, applicant has provided legible markings for the various components appearing in the figure that were deemed to have illegible markings.

In Figure 2, the applicant presents the schematic shown in old FIG. 2 in two separate drawing sheets, Figure 2A and Figure 2B. The two replacement figures show more clearly the details shown in old Figure. 2, and conform to the statutory requirements of 37 C.F.R. § 1.84.

In Figure 3, the applicant marked the program counter units uPC-1 to uPC-4 with their corresponding reference numerals. Thus, uPC-1 is marked with reference numeral 72a (which previously pointed to uPC-4), uPC-2 is marked with reference numeral 72b (which previously pointed to uPC-1), and uPC-3 is marked with reference numeral 72c (which previously pointed to the illustrated multiplexer). Additionally, new reference numeral 72d was added to mark uPC-4.

Attachments following last page of this Amendment:

Replacement Sheet (6 pages)

Annotated Sheet Showing Change(s)
(*originals filed on November 30, 2005*) (3 pages)

REMARKS

The examiner indicated in the May 4, 2006, Advisory Action, that applicant's proposed amendments presented in the April 24, 2006, Amendment in Reply to Final Action of February 24, 2006, were not entered. Accordingly, applicant is re-submitting for the examiner's convenience the proposed amendments, including Amendments to the Claims, Amendments to the Specification, and Amendments to the Drawings, that were presented in the April 24, 2005, Amendment in Reply. Applicant notes that the amendments to the claims include language that was not presented in the April 24, 2005, Amendment in Reply.

The examiner has requested that the Abstract amended in applicant's November 30, 2005, Amendment in Reply to the Action of September 30, 2005 (hereinafter "previous Amendment in Reply") appear on a separate sheet of paper. Attached herewith is the Abstract. The previously made amendments to the Abstract are also presented above, for the examiner's convenience.

Applicant amended the page 10, line 27 of the specification to remove the errant word "that". Applicant thanks the examiner for pointing out the typographical error.

With respect to the examiner's request to replace the reference numeral 16b with 26b on page 3, line 9 of the specification, applicant notes that this amendment was presented in applicant's previous Amendment in Reply. These previously presented amendments are provided above, for the examiner's convenience.

Additionally, attached herewith are the replacement drawing sheets that include the amendments to the drawings previously indicated. Also attached herewith, for the examiner's convenience, are copies of the annotated drawing sheets that were filed on November 30, 2005. The description of the amendments to the drawings, which was submitted in the previous Amendment in Reply is also provided above for the examiner's convenience.

Further, as requested by the examiner, applicant amended claim 7 to remove the second occurrence of word "breakpoint" appearing in the last line of the claim.

The examiner maintained his rejections of claims 1-6, 8-10, 20-25 and 27-29 under 35 U.S.C. §102(e) as being anticipated by U.S. Patent No. 6,058,465 to Nguyen.

The examiner also maintained his rejections of claims 1-6, 8-9, 20-25 and 27-28 under 35 U.S.C. §102(b) as being anticipated by "IA-64 Application Developer's Architecture Guide", May 1999 (hereinafter "Intel").

Additionally, the examiner rejected claims 11-19 and 30-38 under 35 U.S.C. §103(a) as being unpatentable over Nguyen, and rejected claims 10-19, and 29-38 under 35 U.S.C. §103(a) as being unpatentable over Intel. The examiner further rejected claims 7 and 26 under 35 U.S.C. §103(a) over Intel in view of U.S. Patent No. 6,145, 123 to Torrey et al.

Specifically, in response to applicant's previous Amendment in Reply, the examiner stated:

> 57. Applicant argues the novelty/rejection of claim 1 on pages 14-15 of the remarks, in substance that:
>
> "Nowhere, however, does Nguyen disclose that any bits of the VCSR register correspond to particular threads. Therefore, Nguyen neither discloses nor suggests loading data, using the VCHGCR instruction or otherwise, to selected bits according to a processing thread number, as required by applicant's independent claim 1."
>
> "However, nowhere does Intel disclose that data is loaded into selected bits of any register according to a processing thread number."
>
> 58. These arguments are not found persuasive for the following reasons:
> a) Regarding the first argument, every instruction causes some action to occur according to a processing thread number. If thread 1 wants specific data to be loaded into a register, then data is loaded into a register according to thread 1 (thread 1 determines the data to be written). Likewise, if thread 5 wants specific data to be loaded into a register, then that data is loaded according to thread 5 (thread 5 determines everything).
> b) Regarding the second argument, the examiner asserts that applicant is reading the claim too narrowly. Applicant has not defined what a thread number is within the claims. Therefore, a thread number could simply be any number associated with a thread. For instance, in Intel, the thread number is the shift amount in a thread's shift instruction. It is a number associated with a thread. And, data is loaded by a shift instruction according to the shift amount (thread number).
> Applicant must further define thread number to overcome the Intel prior art. (Office Action, pages 16-17)

Applicant amended independent claim 1 to clarify that the data and register are specified by execution of a fast-write instruction, to clarify that the processing thread number is the

number that identifies a corresponding processing thread, and to clarify that bit positions selected in a register are selected according to the processing thread number. Applicant similarly amended independent claim 20, and further amended the claim's preamble to indicate that the instruction that causes a computer to perform the claim's recited operations is a fast-write instruction, and also amended the claim the clarify that the received data and register are specified in the fast-write instruction. As explained in the originally filed application:

> **FAST_WR writes the specified immediate data to the specified FBI CSR. This improves performance by eliminating the need for the FBI unit to pull the data from a transfer register. The FBI unit automatically shifts the immediate data into the appropriate register field corresponding to the thread that is writing the FAST_WR data.**
>
> **A format for the fast write instruction is: fast_wr [immed_data, csr_addr], optional token, where a description of each of the fields follows.**
>
> **The immed_data field represents 10 bits of the immediate data to be written to the control and status register (CSR)" (page 10, lines 20-28 of the originally filed application).**

Thus, which particular bit positions of a register (e.g., a control and status register) are loaded with the received data depends on the particular processing thread number.

Applicant also amended claims 7 and 26 to replace the word "bits" with "bit positions", amended claim 26 to clarify that it is the fast-write instruction that causes the operations recited in those claims to be performed, and amended claims 8 and 27 to recite that the fast-write instruction comprises a token.

As explained in applicant's previous Amendment in Reply, Nguyen, on the other hand, describes a vector processor architecture having vector registers, and corresponding instructions to manipulate operands associated with the vector registers (Abstract). Nguyen's processing architecture uses a vector processor 120 that includes a control register, called VCSR, to control the operation of the processor, and more generally the operation of the processing core 100 (see, for example, col. 5, lines 9-17, and Tables C.4 and C.5 at cols. 27 and 28).

As shown in Table C.5, the VCSR register is a 32-bit register whose various bits enable selection of various processing modes and options. For example, the CBANK bit indicates which of the processor's two 32-vector register banks are to be used. If the CBANK is set to 1,

bank 1 of the vector-registers is selected, whereas if the CBANK bit is cleared (i.e., it is 0), vector-register bank 0 is selected (col. 5, lines 9-12, and col. 28, lines 16-18). As Nguyen discloses, a user can select particular modes of processor operation using Change Vector Register Instruction (VCHGCR) to set those bits that correspond to the desired modes of operation (col. 91-92).

But none of the bits of the VCSR register correspond to specific threads. Therefore, the bit positions of the VCSR register are not selected in accordance with which thread is processing, and thus those bit positions are not selected in accordance with the processing thread number that identifies the processing thread. Accordingly, Nguyen neither discloses nor suggests "receiving data specified by execution of a fast-write instruction in a processing thread identified by a processing thread number; selecting bit positions of a register specified by execution of the fast-write instruction according to the processing thread number; and loading the data into the selected bit positions of the register," as required by applicant's independent claim 1.

As also explained in applicant's previous Amendment in Reply, Intel describes an instruction set for the IA-64 processing architecture. One instruction used for in IA-64 architecture is the Shift Left ("*shl*") instruction. As explained in Intel's page 7-165, execution of the "*shl*" instruction causes the value stored in source register r_2 to be left shifted by an amount specified by register r_3, or alternatively by a numerical value specified as a parameter of the instruction. Bit positions vacated by the shifted value in register r_2 are filled with zeros. The shifted value in register r_2 is then loaded into a specified target register r_1 (as indicated by one of the parameters of the *shl* instruction). Although the IA-64 supports multiple thread execution, the use of threads does not affect the operation of the *shl*, or for that matter the other instructions referred to by the examiner on Intel's page C-23.

The examiner contends that "[a]pplicant has not defined what a thread number is within the claims. Therefore, a thread number could simply be any number associated with a thread. For instance, in Intel, the thread number is the shift amount in a thread's shift instruction. It is a number associated with a thread." (Office Action, page 16, paragraph 58(b)). However, claim 1 recites that the thread number is the number that identifies the thread. Applicant's processing

thread number is different from the instruction's shift operand, described in the Intel reference, which specifies the extent of the shift to be performed by the instruction.

Thus, Intel does not disclose or suggest "receiving data specified by execution of a fast-write instruction in a processing thread identified by a processing thread number; selecting bit positions of a register specified by execution of the fast-write instruction according to the processing thread number; and loading the data into the selected bit positions of the register," as required by applicant's independent claim 1.

Since neither Nguyen nor Intel discloses or suggests, alone or in combination at least the features of "receiving data specified by execution of a fast-write instruction in a processing thread identified by a processing thread number; selecting bit positions of a register specified by execution of the fast-write instruction according to the processing thread number; and loading the data into the selected bit positions of the register," applicant's independent claim 1 is patentable over the art cited by the examiner. Claims 2-19 depend from independent claim 1 and are therefore patentable for at least the same reasons as independent claim 1.

Independent claim 20 recites the features of "receive data specified in the fast-write instruction in a processing thread identified by a processing thread number specified in the fast-write instruction; select bit positions of a register specified in the fast-write instruction according to the processing thread number; and load the data into the selected bit positions of the register." For reasons similar to those provided with respect to independent claim 1, at least these features are not disclosed by the cited art. Accordingly, independent claim 20 is patentable over the cited art. Claims 21-38 depend from independent claim 20 and are therefore patentable for at least the same reasons as independent claim 20.

It is believed that all the rejections and/or objections raised by the examiner have been addressed.

In view of the foregoing remarks, applicant respectfully submits that the application is in condition for allowance and such action is respectfully requested at the examiner's earliest convenience.

All of the dependent claims are patentable for at least the reasons for which the claims on which they depend are patentable.

Canceled claims, if any, have been canceled without prejudice or disclaimer.

Any circumstance in which the applicant has (a) addressed certain comments of the examiner does not mean that the applicant concedes other comments of the examiner, (b) made arguments for the patentability of some claims does not mean that there are not other good reasons for patentability of those claims and other claims, or (c) amended or canceled a claim does not mean that the applicant concedes any of the examiner's positions with respect to that claim or other claims.

Please apply any charges or credits to deposit account 06-1050, referencing attorney docket 10559-308US1.

Respectfully submitted,

Date: June 26, 2006

Ido Rabinovitch
Attorney for Intel Corporation
Reg. No. L0080

Fish & Richardson P.C.
Telephone: (617) 542-5070
Facsimile: (617) 542-8906
21365144.doc